MANUAL BÁSICO DE MATLAB



Imagen y Acústica - Comisión: Com D - Turno: Noche

INTRODUCCIÓN

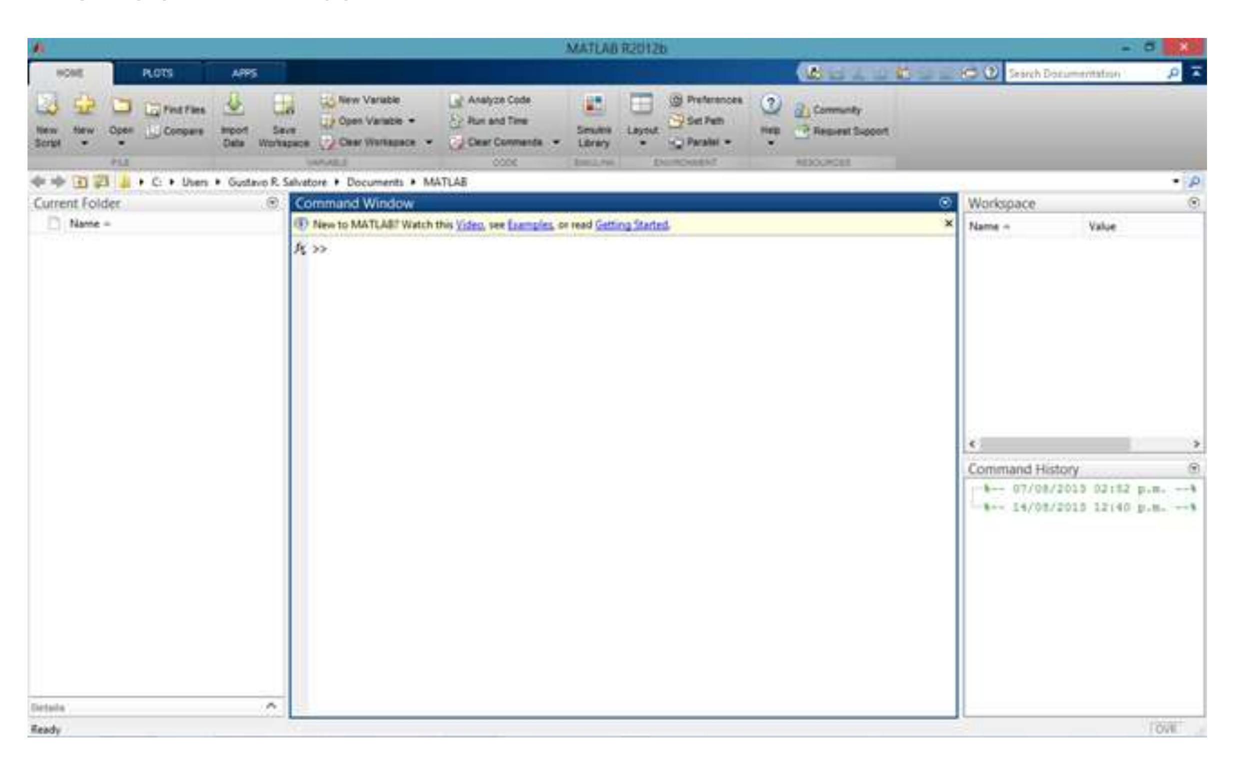
MATLAB es el nombre abreviado de “**MAT**riz **LAB**oratory”. Es un programa para realizar cálculos numéricos con vectores y matrices, y por lo tanto se puede trabajar también con números escalares (tanto reales como complejos), con cadenas de caracteres y con otras estructuras de información más complejas.

Matlab es un lenguaje de alto rendimiento para cálculos técnicos, es al mismo tiempo un entorno y un lenguaje de programación. Uno de sus puntos fuertes es que permite construir nuestras propias herramientas reutilizables. Podemos crear fácilmente nuestras propias funciones y programas especiales (conocidos como M-archivos) en código Matlab, los podemos agrupar en Toolbox (también llamadas librerías): colección especializada de M-archivos para trabajar en clases particulares de problemas.

Matlab, a parte del cálculo matricial y álgebra lineal, también puede manejar polinomios, funciones, ecuaciones diferenciales ordinarias, gráficos.

CARACTERÍSTICAS BÁSICAS

EL ESPACIO DE TRABAJO DE MATLAB

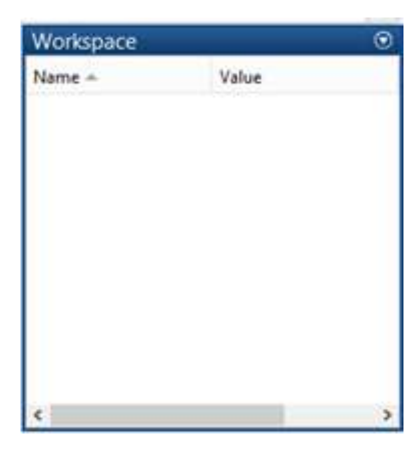


Todas las sentencias que vamos a utilizar las escribiremos en la ventana **Command Window** (ventana de comandos). Es la ventana de mayor tamaño.

Si se quiere obtener información acerca de las variables que estamos utilizando podemos verlas en la ventana Workspace (espacio de trabajo) o usar:

who para obtener la lista de las variables (no de sus valores)

whos para obtener la lista de las variables e información del tamaño, tipo y atributos (tampoco da valores)



Si lo que queremos obtener es el valor que tiene una variable lo hacemos escribiendo el nombre de la variable y pulsando **Intro**.

Para recordar órdenes previas usamos las flechas del teclado ↑ y ↓. También podemos verlas en la ventana **Command History**, ventana situada en la parte inferior derecha:



MATEMÁTICA BASICA

Matlab ofrece la posibilidad de realizar las siguientes operaciones básicas:

Operación	Símbolo	Expresión en Matlab
suma	+	a + b
resta	-	a - b
multiplicación	*	a * b
división	/	a / b
potencia	^	a ^ b

El orden de precedencia es:

Orden de precedencia de operaciones	
1°	^
2°	* /
3°	+ -

Matlab no tiene en cuenta los espacios.

Si queremos que evalúe la línea pero que no escriba la respuesta, basta escribir punto y coma (;) al final de la sentencia.

Si la sentencia es demasiado larga para que entre en una sola línea podemos poner tres puntos (…) seguido de la tecla enter para indicar que continúa en la línea siguiente.

Ejemplos:

```
>> a = 2        % damos valor a la variable a y la escribe por pantalla

a =  2

>> b = 3;       % no escribe el valor de b por el; del final

>> a + b        % realiza la suma de dos variables y guarda la solución en la variable ans

ans = 5

>> a / b

ans =  0.6667

>> a ^ b

ans =  8

>> 3.5 * a

ans =  7
```

```
>> who  % da una lista de los nombres de las variables usadas

Your variables are:

a  ans  b

>> whos  % da una lista de las variables usadas más completa que la anterior

  Name      Size      Bytes  Class     Attributes

    a       1x1           8  double

  ans       1x1           8  double

    b       1x1           8  double
```

ALMACENAR Y RECUPERAR DATOS

Matlab permite guardar y cargar datos de los archivos del computador.

En el menú File, la opción **Save Workspace as** guarda todas las variables actuales e **Import Data** carga variables de un espacio de trabajo guardado previamente.

Otra forma de realizarlo sería guardar el estado de una sesión de trabajo con el comando **save** antes de salir:

```
>> save
```

Al teclear esto, automáticamente se crea un fichero llamado matlab.mat. Puede recuperarse la siguiente vez que se arranque el programa con el comando load:

```
>> load
```

FORMATOS DE VISUALIZACIÓN DE NÚMEROS

Matlab no cambia la representación interna de un número cuando se escogen distintos formatos, sólo se modifica la forma de visualizarlo.

Tipo	Resultado	Ejemplo: >> pi
format short	Formato coma fija con 4 dígitos después de la coma (es el formato que viene por defecto)	3.1416
format long	Formato coma fija con 14 o 15 dígitos después de la coma	3.14159265358979
format short e	Formato coma flotante con 4 dígitos después de la coma	3.1416e+000
format long e	Formato coma flotante con 14 o 15 dígitos después de la coma	3.141592653589793e+000
format short g	La mejor entre coma fija o flotante con 4 dígitos después de la coma	3.1416
format long g	La mejor entre coma fija o flotante con 14 o 15 dígitos después de la coma	3.14159265358979
format short eng	Notación científica con 4 dígitos después de la coma y un exponente de 3	3.1416e+000
format long eng	Notación científica con 16 dígitos significantes y un exponente de 3	3.14159265358979e+000
format bank	Formato coma fija con 2 dígitos después de la coma	3.14
format hex	Hexadecimal	400921fb54442d18
format rat	Aproximación racional	355/113
format +	Positivo, negativo o espacio en blanco	+

ACERCA DE LAS VARIABLES

Matlab almacena el último resultado obtenido en la variable ans.

Las variables son sensibles a las mayúsculas, deben comenzar siempre con una letra, no pueden contener espacios en blanco y pueden nombrarse hasta con 63 caracteres (en versiones anteriores no permitía tantos caracteres). Si se nombra una variable con más de 63 caracteres truncará el nombre de dicha variable.

Algunas variables especiales de Matlab:

Variable	Definición	Valor
ans	Variable usada por defecto para almacenar el último resultado	? ? ?
pi	Razón de una circunferencia a su diámetro	3.1416
eps	Número más pequeño, tal que cuando se le suma 1, crea un número en coma flotante en el computador mayor que 1	2.2204e-016
inf	Infinito	Inf
nan	Magnitud no numérica	NaN
i y j	$i = j = \sqrt{-1}$	0 + 1.0000i
realmin	El número real positivo más pequeño que es utilizable	2.2251e-308
realmax	El número real positivo más grande que es utilizable	1.7977e+308

Tecleando **clear** podemos borrar todas las variables del espacio de trabajo, pero no borra lo de las demás ventanas, es decir, no desaparece lo que hay escrito en la ventana de comandos.

Tecleando **clc** borramos lo que hay en la ventana de comandos pero no borra las variables de la memoria del espacio de trabajo.

Algunos comandos de Matlab nos facilitan información sobre la fecha, como **clock**, **date** o **calendar**.

OTRAS CARACTERÍSTICAS BÁSICAS

Los comentarios se escriben después del símbolo de por ciento (%), de este modo todo lo que se escriba a continuación en la misma línea no será leído por Matlab. Podemos colocar varias órdenes en una línea si se separan correctamente, puede ser:

por comas (,) que hacen que se visualicen los resultados

o puntos y comas (;) que suprimen la impresión en pantalla

FUNCIONES MATEMÁTICAS COMUNES

APROXIMACIONES

Función	¿Qué hace?	Ejemplo x = 5.92
ceil (x)	redondea hacia infinito	6
fix (x)	redondea hacia cero	5
floor (x)	redondea hacia menos infinito	5
round (x)	redondea hacia el entero más próximo	6

(con una **x** escalar, vector o matriz, redondea en cada caso los elementos individualmente)

Ejemplo:

```
>> round ( [19.8932 17.256 -2.0565 0.98] )

ans =     20    17    -2     1
```

TRIGONOMETRÍA

Función	¿Qué hace?
… (x)	función trigonométrica con el ángulo expresado en radianes
sin (x) **cos (x)** **tan (x)** **csc (x)** **sec (x)** **cot (x)**	seno (radianes) coseno tangente cosecante secante cotangente
…d (x)	función trigonométrica con el ángulo expresado en grados
sind (x) …	seno (grados) …
…h (x)	función trigonométrica hiperbólica con el ángulo expresado en radianes
sinh (x) …	seno hiperbólico (radianes) …
a… (x)	inversa de la función trigonométrica con el resultado expresado en radianes
asin (x) …	arco seno (radianes) …
a…d (x)	inversa de la función trigonométrica con el resultado expresado en grados
asind (x) …	arco seno (grados) …
a…h (x)	inversa de la función trigonométrica hiperbólica con el resultado expresado en radianes
asinh (x) …	arco seno hiperbólico (radianes) …

Ejemplos:

```
>> sin (pi/2)

ans =     1

>> sind (-90)

ans =    -1
```

MAS OPERACIONES

Función	¿Qué hace?
abs (x)	valor absoluto o magnitud de un número complejo
sign (x)	signo del argumento si x es un valor real (-1 si es negativo, 0 si es cero, 1 si es positivo)
exp (x)	exponencial
gcd (m,n)	máximo común divisor
lcm (m,n)	mínimo común múltiplo
log (x)	logaritmo neperiano o natural
log2 (x)	logaritmo en base 2
log10 (x)	logaritmo decimal
mod(x,y)	módulo después de la división
rem (x,y)	resto de la división entera
sqrt (x)	raíz cuadrada
nthroot (x,n)	raíz n-ésima de x

(x e y cualquier escalar, m y n enteros)

Ejemplos:

```
>> abs (-3)      % valor absoluto de -3

ans =     3

>> sign (-10)    % signo del número -10

ans =     -1

>> gcd (9,12)    % máximo común divisor entre 9 y 12

ans =     3
```

VECTORES Y MATRICES

Para crear un vector introducimos los valores deseados separados por espacios (o comas) todo ello entre corchetes [ ]. Si lo que queremos es crear una matriz lo hacemos de forma análoga pero separando las filas con puntos y comas (;).

Generalmente usaremos letras mayúsculas cuando nombremos a las matrices y minúsculas para vectores y escalares.

Ejemplos:

```
>> x = [1 0 -1 5 -6]      % es un vector, los elementos los separamos con espacios

x =    1   0   -1   5   -6

>> y = [6,1,6,1]          % es otro vector, los elementos los separamos con comas

y =    6   1   6   1

>> A = [1 2 3; 4 5 6]     % es una matriz con 2 filas y 3 columnas

A =    1   2   3    4   5   6
```

CONSTRUCCIÓN ABREVIADA DE ALGUNOS VECTORES

Se pueden crear usando las siguientes sentencias:

(a:b) crea un vector que comienza en el valor a y acaba en el valor b aumentando de 1 en 1.

(a:c:b) crea un vector que comienza en el valor a y acaba en el valor b aumentando de c en c.

linspace (a,b,c) genera un vector linealmente espaciado entre los valores a y b con c elementos.

linspace (a,b) genera un vector linealmente espaciado entre los valores a y b con 100 elementos.

logspace (a,b,c) genera un vector logarítmicamente espaciado entre los valores 10^a y 10^b con c elementos.

logspace (a,b) genera un vector logarítmicamente espaciado entre los valores 10^a y 10^b con 50 elementos.

Ejemplos:

```
>> (1:4)              % crea un vector que comienza en 1, aumenta de 1 en 1 y acaba en 4

ans =    1    2    3    4

>> (1:4:10)           % comenzando en 1, aumenta de 4 en 4 hasta el 10 y por eso acaba en 9

ans =    1    5    9

>> linspace (2,6,3)   % genera un vector desde el 2 al 6 con 3 elementos equidistantes

ans =    2    4    6

>> linspace (2,6,4)   % genera un vector desde el 2 al 6 con 4 elementos equidistantes

ans =    2.0000   3.3333   4.6667   6.0000

>> logspace (0,2,4)   % genera un vector logarítmicamente espaciado entre 10^0 y 10^2 con 4
                      elementos

ans =    1.0000   4.6416   21.5443   100.0000
```

CONSTRUCCIÓN DE ALGUNAS MATRICES

Al igual que pasa con los vectores, existen sentencias que nos ayudan a crear más rápidamente algunas matrices que Matlab ya tiene predefinidas (m y n deben tomar valores naturales):

zeros (n) crea una matriz cuadrada n x n de ceros.

zeros (m,n) crea una matriz m x n de ceros.

ones (n) crea una matriz cuadrada n x n de unos.

ones (m,n) crea una matriz m x n de unos.

rand (n) crea una matriz cuadrada n x n de números aleatorios con distribución uniforme (0,1).

rand (m,n) crea una matriz m x n de números aleatorios con distribución uniforme (0,1).

randn (n) crea una matriz cuadrada n x n de números aleatorios con distribución normal (0,1).

randn (m,n) crea una matriz m x n de números aleatorios con distribución normal (0,1).

eye (n) crea una matriz cuadrada n x n de unos en la diagonal y ceros el resto.

eye (m,n) crea una matriz m x n de unos en la diagonal y ceros el resto.

magic (n) crea una matriz cuadrada n x n de enteros de modo que sumen lo mismo las filas y las columnas.

Ejemplos:

```
>> zeros (3)      % matriz cuadrada 3 x 3 de ceros

ans =     0    0    0

          0    0    0

          0    0    0

>> eye (2)        % matriz  identidad o unidad

ans =     1    0

          0    1
```

OPERACIONES BÁSICAS CON MATRICES

Símbolo	Expresión	Operación
+	A + B	Suma de matrices
-	A - B	Resta de matrices
*	A * B	Multiplicación de matrices
.*	A .* B	Multiplicación elemento a elemento de matrices
/	A / B	División de matrices por la derecha
./	A ./ B	División elemento a elemento de matrices por la derecha
\	A \ B	División de matrices por la izquierda
.\	A .\ B	División elemento a elemento de matrices por la izquierda
^	A ^ n	Potenciación (n debe ser un número, no una matriz)
.^	A .^ B	Potenciación elemento a elemento de matrices
'	A '	Trasposición compleja conjugada
.'	A .'	Trasposición de matrices

FUNCIONES PARA OPERAR CON VECTORES

Función	¿Qué hace?
cross (x,y)	producto vectorial entre los vectores x e y
dot (x,y)	producto escalar entre los vectores x e y

FUNCIONES PARA EL ANÁLISIS DE MATRICES

Función	¿Qué hace?
cond (A)	número de condición
det (A)	determinante
diag (v)	crea una matriz diagonal con el vector v sobre la diagonal
diag (A)	extrae la diagonal de la matriz A como un vector columna
eig (A)	valores propios
inv (A)	matriz inversa
length (A)	máxima dimensión
norm (A)	norma
norm (A,n)	norma-n
normest (A)	estimación de la norma-2
null (A)	espacio nulo
orth (A)	ortogonalización
pinv (A)	pseudoinversa
poly (A)	polinomio característico
rank (A)	rango
rref (A)	reducción mediante la eliminación de Gauss de una matriz
size (A)	dimensiones
trace (A)	traza
tril (A)	matriz triangular inferior a partir de la matriz A
triu (A)	matriz triangular superior a partir de la matriz A

(Con A matriz, v vector y n número natural)

OTRAS OPERACIONES CON MATRICES

Función	¿Qué hace?
find (A)	devuelve los índices donde las entradas de A son distinto de cero
fliplr (A)	intercambia la matriz de izquierda a derecha
flipud (A)	intercambia la matriz de arriba a abajo
reshape (A,m,n)	devuelve una matriz m x n cuyos elementos se toman por columnas de A, si A no contiene m x n elementos daría un error
rot90 (A)	gira la matriz 90° en sentido contrario a las agujas del reloj
rot90 (A,n)	gira la matriz n x 90°
expm (A)	matriz exponencial
logm (A)	matriz logarítmica
sqrtm (A)	matriz de raíces cuadradas
funm (A,@función)	evalúa la función que indiquemos en la matriz A
exp, log, sqrt…	operan elemento a elemento
[VE,VA] = eig (A)	VE son los vectores propios y VA son los valores propios
[L,U] = lu (A)	factorización LU
[Q,R] = qr (A)	factorización QR

(Con A matriz, m y n naturales)

OPERACIONES RELACIONALES Y LÓGICAS

Como entradas a las expresiones relacionales y lógicas, Matlab considera que cero es falso y que cualquier número distinto de cero es verdadero. La salida de expresiones de este tipo produce 1 si es verdadero y 0 si es falso.

OPERADORES RELACIONALES

Operador	¿Qué significa?
<	menor que
<=	menor o igual que
>	mayor que
>=	mayor o igual que
==	igual a
~=	distinto de

La salida de las operaciones lógicas se puede utilizar también en operaciones matemáticas.

OPERADORES LÓGICOS

Operador	¿Qué significa?
&	y
|	o
~	no

Además de los operadores relacionales y lógicos básicos anteriores, Matlab proporciona una serie de funciones relacionales y lógicas adicionales que incluyen:

Función	¿Qué significa?
xor (x,y)	operación "o" exclusiva, devuelve 0 si ambas son falsas o ambas verdaderas y devuelve 1 si una es falsa y la otra verdadera
any (x)	devuelve 1 si algún elemento en un vector x es no nulo y devuelve 0 si son todos nulos, si se trata de una matriz da una respuesta por cada columna
all (x)	devuelve 1 si todos los elementos en un vector x son no nulos y 0 si existe alguno nulo y si se trata de una matriz da una respuesta por cada columna
exist ('x')	devuelve uno si existe y cero si no existe
isnan (x)	devuelve unos en magnitudes no numéricas (NaN) en x
isinf (x)	devuelve unos en magnitudes infinitas (Inf) en x
isfinite (x)	devuelve unos en valores finitos en x

Podemos ver muchos más casos pero todos serían similares: ischar, isempty, isequal, isfloat, isinteger, islogical, isnumeric, isprime, isreal, isscalar, isspace, etc.

Existe un orden de precedencia para operadores aritméticos, lógicos y relacionales, en la siguiente tabla van de mayor a menor precedencia:

Orden de precedencia de operadores							
1°	^	.^	'	.'			
2°	*	/	\	.*	./	.\	
3°	+	-	~	+(unario)	-(unario)		
4°	:	>	<	>=	<=	==	~=
5°	|	&					

Ejemplos:

```
>> a = 1:5, b = 5-a       % definimos dos vectores

a =    1    2    3    4    5

b =    4    3    2    1    0

>> r1 = a<5     % pregunta si a es menor que 5, devuelve 1 cuando es verdadero y 0 cuando es falso

r1 =    1    1    1    1    0

>> r2 = a==b    % pregunta si a es igual a b, devuelve 1 cuando es verdadero y 0 cuando es falso

r2 =    0    0    0    0    0

>> r3 = a~=b    % pregunta si a es distinto a b, devuelve 1 cuando es verdadero y 0 cuando es falso

r3 =    1    1    1    1    1

>> r4 = (a>b)&(b>-3)     % pregunta si a>b y b>-3, devuelve 1 cuando es verdadero y 0 cuando es
                          falso

r4 =    0    0    1    1    1

>> c = [Inf 0 5 -8 NaN 94];

>> exist ('c')     % pregunta si existe alguna variable llamada c

ans =    1

>> isnan (c)     % pregunta cuando c es NaN, devuelve 1 cuando es verdadero y 0 cuando es falso

ans =    0    0    0    0    1    0

>> isinf (c)     % pregunta cuando c es Inf, devuelve 1 cuando es verdadero y 0 cuando es falso

ans =    1    0    0    0    0    0

>> isfinite (c)     % pregunta cuando c es finito, devuelve 1 cuando es verdadero y 0 cuando es falso

ans =    0    1    1    1    0    1
```

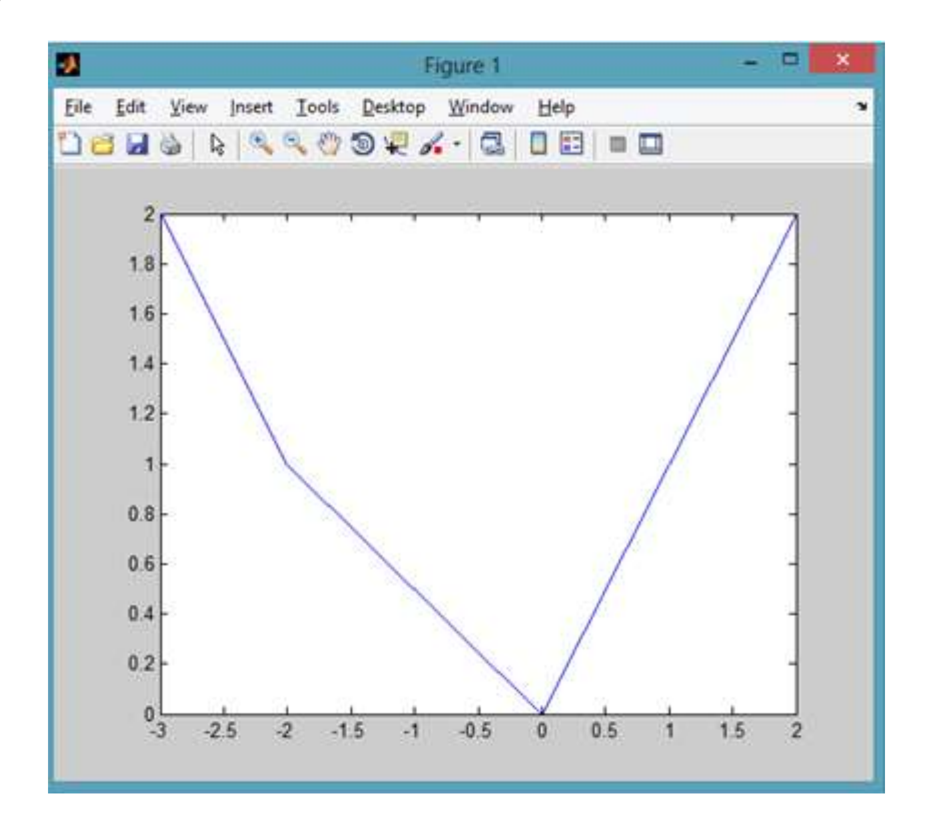
GRÁFICAS 2-D

La orden plot genera una gráfica. Los argumentos deben ser vectores de la misma longitud.

Ejemplo:

```
>> x = [-3 -2 0 1 2]; y = [2 1 0 1 2];

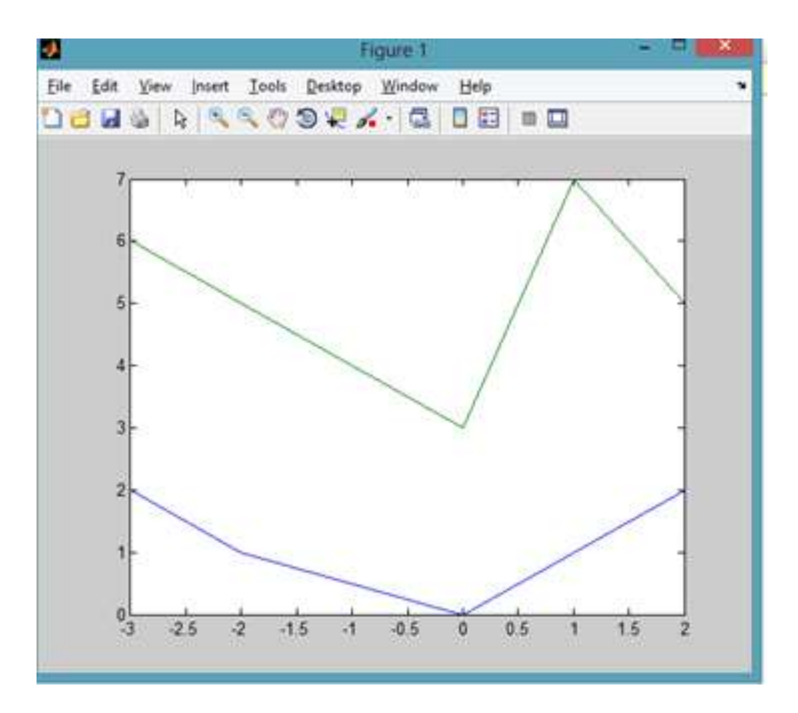
>> plot (x,y)
```



La función plot nos permite otras opciones como superponer gráficas sobre los mismos ejes:

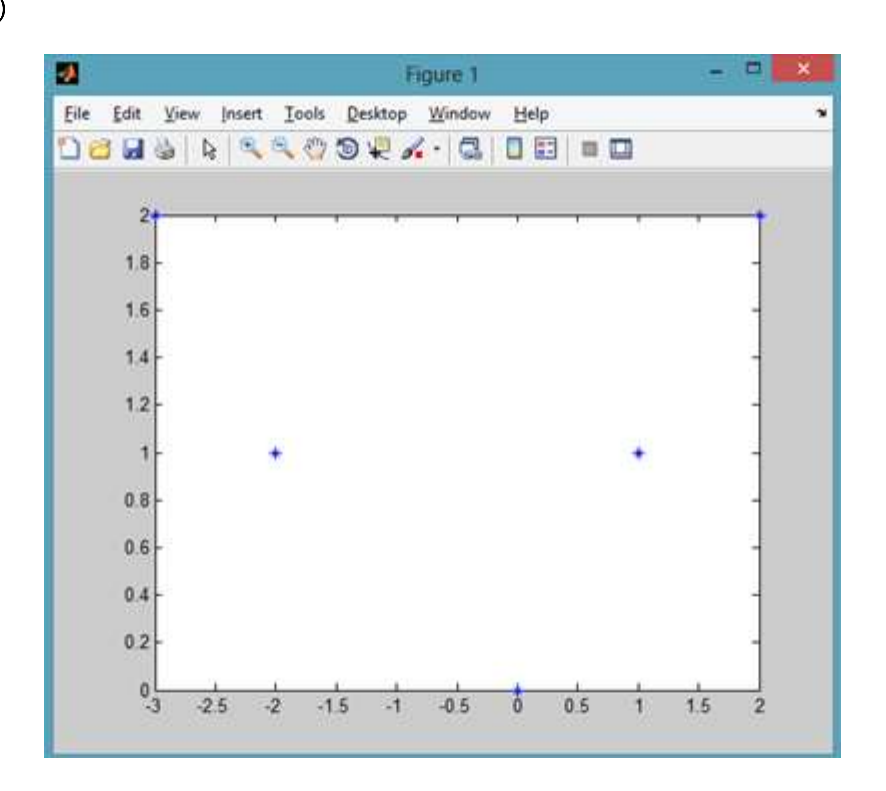
>> x = [-1 -1 0 1 2]; y = [2 1 0 1 2]; z = [6 5 3 7 5];

>> plot (x,y,x,z)



También podemos usar distintos tipos de líneas para el dibujo de la gráfica:

>> plot (x,y,'*')



Además podemos colocar etiquetas o manipular la gráfica:

etiqueta sobre el eje X de la gráfica actual:	>> **xlabel('texto')**
etiqueta sobre el eje Y de la gráfica actual:	>> **ylabel('texto')**
título en la cabecera de la gráfica actual:	>> **title('texto')**
texto en el lugar especificado por las coordenadas:	>> **text(x,y, 'texto')**
texto, el lugar lo indicamos después con el mouse:	>> **gtext('texto')**
dibujar una grilla:	>> **grid**
fija valores máximo y mínimo de los ejes:	>> **axis([xmin xmax ymin ymax])**
fija que la escala en los ejes sea igual:	>> **axis equal**
fija que la gráfica sea un cuadrado:	>> **axis square**
desactiva axis equal y axis square:	>> **axis normal**
abre una ventana de gráfico:	>> **hold on**
borra lo que hay en la ventana de gráfico:	>> **hold off**

Todas estas órdenes se las podemos dar desde la propia ventana de la gráfica.

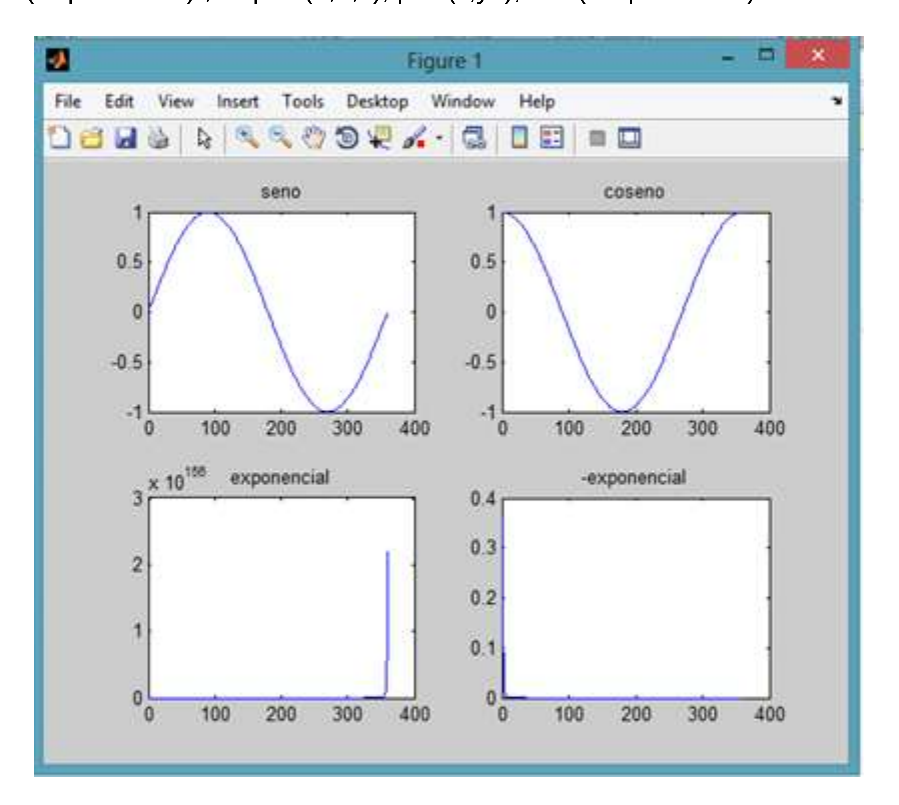
Otros comandos relacionados con las gráficas son los siguientes:

Orden	¿Qué hace?	Imagen
area	colorea el area bajo la gráfica	
bar	diagrama de barras (verticales)	
barh	diagrama de barras (horizontales)	
hist	histograma	
pie	sectores	
rose	histograma polar	
stairs	gráfico de escalera	
stem	secuencia de datos discretos	
loglog	como plot pero con escala logarítmica en ambos ejes	
semilogx	como plot pero escala logarítmica en el eje x	
semilogy	como plot pero escala logarítmica en el eje y	

Una ventana gráfica se puede dividir en m particiones horizontales y en n verticales, de modo que cada subventana tiene sus propios ejes, y para hacer esto vamos a usar subplot (m,n,p) donde p indica la subdivisión que se convierte en activa.

>> x = 1:360; y1 = sind (x); y2 = cosd (x); y3 = exp (x); y4 = exp (-x);

>> subplot (2,2,1), plot (x,y1), title ('seno') ;subplot (2,2,2), plot (x,y2), title ('coseno'); subplot (2,2,3), plot (x,y3), title ('exponencial') ;subplot (2,2,4), plot (x,y4), title ('-exponencial')



Para volver al modo por defecto basta escribir: subplot (1,1,1).

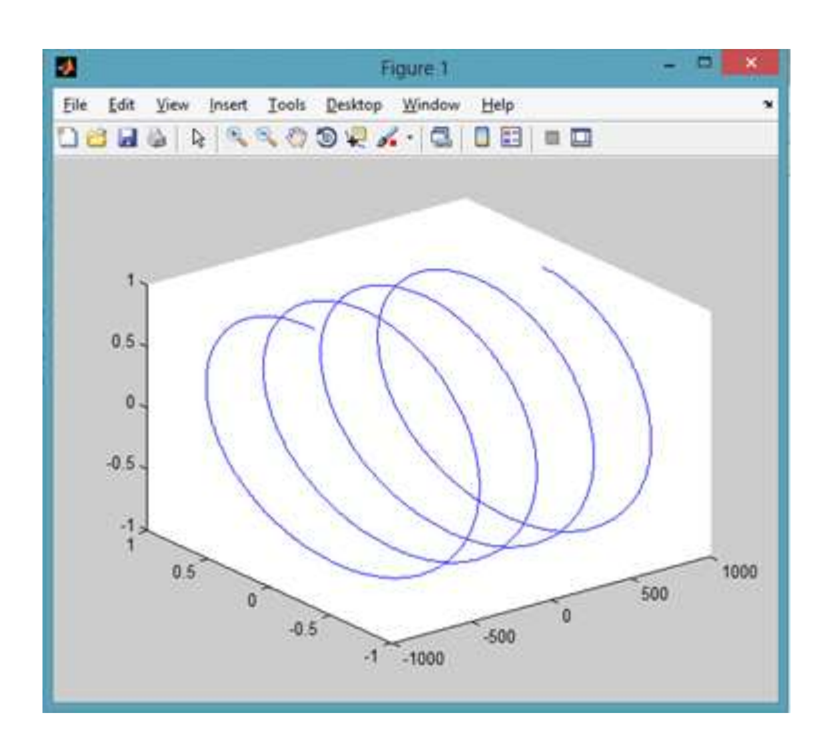
GRÁFICAS 3-D

Gráficos de línea:

También podemos crear gráficas en 3 dimensiones, se trata de extender la orden de plot (2-D) a plot3 (3-D) donde el formato será igual pero los datos estarán en tripletes:

>> x = -720:720; y = sind (x); z = cosd (x);

>> plot3 (x,y,z)



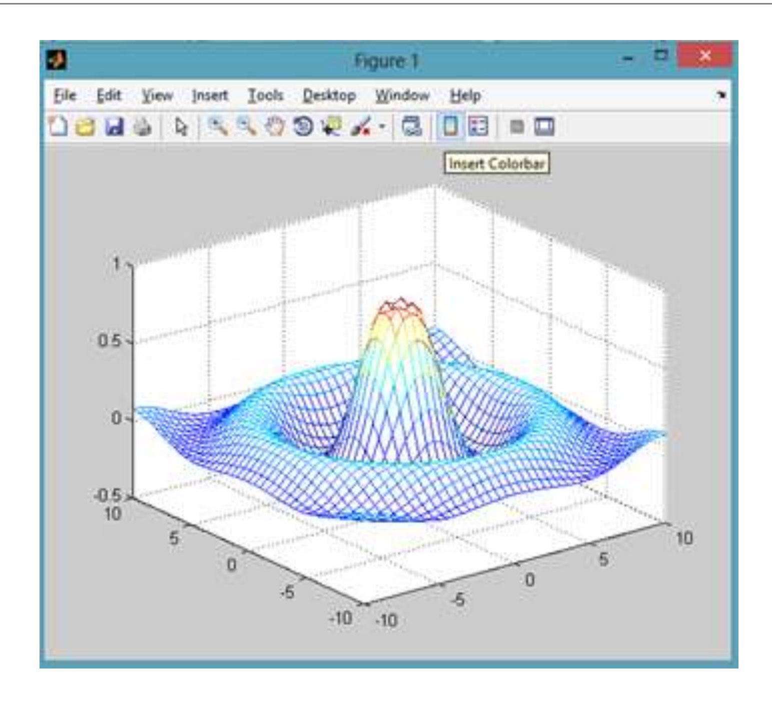
Ejemplo:

>> x = -10:0.5:10; y = -10:0.5:10;

>> [X,Y] = meshgrid (x,y);   % crea matrices para hacer la malla

>> Z = sin (sqrt (X .^2 + Y .^2)) ./ sqrt (X .^ 2 + Y .^ 2 + 0.1);

>> mesh (X,Y,Z)   % dibuja la gráfica



Hubiera dado igual si hubiéramos escrito:

>> [X,Y] = meshgrid (-10:0.5:10);

>> Z = sin (sqrt (X .^2 + Y .^ 2)) ./ sqrt (X .^ 2 + Y .^ 2 + 0.1);

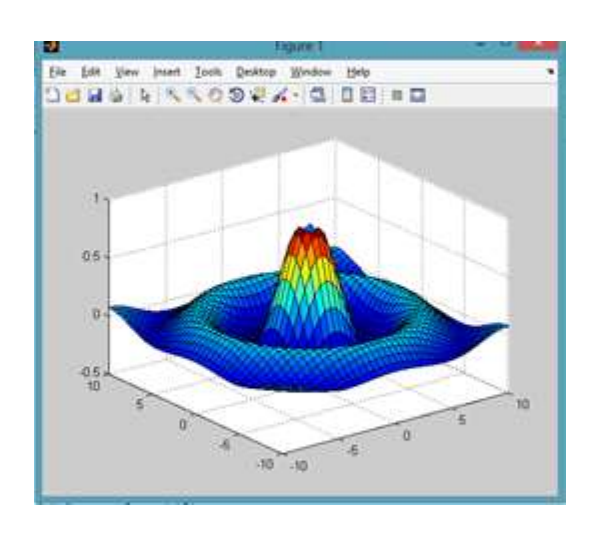
>> mesh (X,Y,Z)

Gráfica de superficie:

Es similar a la gráfica de malla, pero aquí se rellenan los espacios entre líneas. La orden que usamos es surf con los mismos argumentos que para mesh.

Ejemplo:

>> surf (X,Y,Z)



POLINOMIOS

RAÍCES

Un polinomio se representa por un vector fila con sus coeficientes en orden descendiente, no debemos olvidar colocar los términos con coeficiente nulo.

Así por ejemplo si queremos indicar el polinomio $x^3 + x^2 + 2$ escribiríamos [1 1 0 2].

Para encontrar las raíces de un polinomio **p** usaremos la función **roots (p)**.

Si conocemos las raíces de un polinomio es posible construir el polinomio asociado mediante la función **poly (r).**

Matlab trabaja con los polinomios como vectores fila y con las raíces como vectores columnas.

CARACTERISTICAS

Función	¿Qué es?
conv (p,q)	multiplica los dos polinomios p y q
deconv (c,q)	divide el polinomio c entre q
polyder (p)	calcula la derivada del polinomio p
polyder (p,q)	calcula la derivada del producto de los polinomios p y q
polyval (p,A)	evalúa el polinomio p en todos los valores de la matriz A

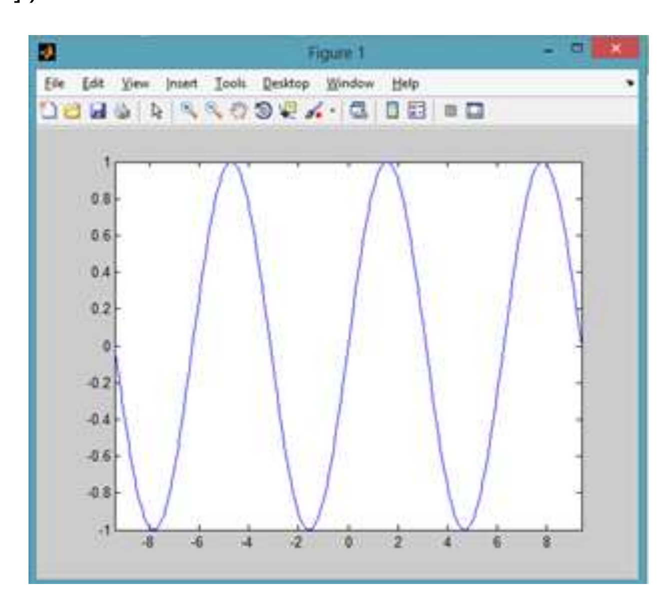
ANÁLISIS NUMÉRICO

REPRESENTACIÓN GRÁFICA

Existe la función fplot que evalúa la función que se desea representar en la gráfica de salida. Como entrada, necesita conocer el nombre de la función como una cadena de caracteres y el rango de representación como un vector de dos elementos: fplot ('nombre', [valor min, valor max]).

Ejemplo:

>> fplot ('sin', [-3*pi,3*pi])



CARACTERISTICAS

Función	¿Qué hace?
diff ('f')	derivada de la función respecto a x
diff ('f',t)	derivada parcial de la función respecto a t
diff ('f',n)	derivada n-ésima de la función respecto a x
feval ('f',a)	evalúa la función en a
fminbnd ('f',a,b)	calcula el mínimo de una función de una variable
fzero ('f',a)	busca el cero de una función unidimensional f más próximo al punto a
quad ('f',a,b)	aproxima la integral definida (según la cuadratura de Simpson)
trapz (x,y)	integral numérica trapezoidal de la función formada al emparejar los puntos de los vectores x e y

(f función, n número natural, a y b valores numéricos, x e y vectores del mismo tamaño)

Ejemplos:

> diff ('sin (7*x) ')   % derivada respecto a x

ans = 7*cos(7*x)

>> diff ('(exp (x) * cos (3*x*y))','y')   % derivada parcial respecto a y

ans = -3*exp(x)*sin(3*x*y)*x